REMARKS

The Applicant would like to thank the Examiner for her attention to this Application.

Responsive to the Examiner's objection to the Abstract, please replace the Abstract with the following:

ABSTRACT

A method and apparatus of dynamically managing non-volatile memory items on a wireless device through a network, the method comprising the steps of: when connecting to the network, checking for a unique identifier item stored in the non-volatile memory items; if the unique identifier item exists, checking whether a value stored in the unique identifier is the same as a software identifier located in software on the wireless device; if the unique identifier item does not exist or the value is different from the software identifier, sending the software identifier along with an identifier indicating a carrier to the network along with an identifier indicating a carrier; receiving from the network a set of changes relates to the software identifier; executing the set of changes to update the non-volatile memory items; and writing the software identifier to the unique identifier item.

Responsive to the Examiner's objections to the application because the applicant fails to disclose "SIM/RUIM" number 144, please insert the following paragraph on page 6 and a new paragraph after line 3:

> A mobile station may require a removable user identity module (RUIM) or a subscriber identity module (SIM) card in order to operate in a CDMA network. The SIM/RUIM interface 144 is normally similar to a card-slot into which a SUIM/RUIM card can be inserted and ejected like a diskette or a PCMCIA card.

Response to the Examiner's objections to claims 4 and 7, the applicant submits herewith amended claims 4 and 7 in which the typographical error for the word "previous" has been

corrected in claim 4 and in which the term "NV" has been changed to "non-volatile memory". Further, claim 7 has been further amended to change the word "policies" to "schemes" as found on page 10, line 1.

Responsive to the Examiner's provisional rejection based on double patenting, the applicant submits that changes made to corresponding Application No. 10/765,512 now distinguish the claims of that co-pending application from the claims of the present application. The applicant therefore submits that the provisional obviousness-type double patenting rejection is now overcome.

Responsive to the Examiner's objections to claim 10a, the applicant has now amended claim 10a to indicate that the avoiding step should avoid non-volatile memory items created under traditional management. This is specified in the present specification on page 10 under the heading "Rule 1". The applicant submits that the Examiner's objections to claims 7 and 10a are now overcome based on the above submissions.

The applicant has further modified claims 1 and 10a to add an otherwise statement to complete the if-else statement as required by the Examiner's objection in paragraph 14.

The Examiner has objected to claims 1, 3 to 10b as being anticipated by U.S. Publication No. 2003/0221189 to Birum et al. Based on the amendments and submissions herein, the applicant submits that the application now overcomes the Birum reference.

The present application relates to dynamic management of persistent data in the non-volatile memory of a wireless device. It is often that a new software load is added to the device and that the new software loader requires persistent data stored in the non-volatile memory to be updated so it matches the new load. However, the situation becomes more complicated as new software loads may modify persistent data during the operation of the wireless device. For example, when the user changes device settings or enters user specific data which overrides default values initially set by the new software load.

Birum teaches a method and system of transparently changing versions of an application on a client. This is done by updating versions of resources for the application. The term "resource" is defined as any data an application uses for execution, and includes a particular portion of a file. The file may be a data file, a dynamic link library and executable program, a component and the like [paragraph 19].

Purge lists are used for upgrading and/or rolling back and are treated and stored for every version available or supported.

The applicant submits herewith-amended claim 1. Claim 1 has been amended to correct an antecedent error by including the word "item" after "unique identifier", and further, by adding the second part of the if-else statement.

The applicant respectfully disagrees that claim 1 is anticipated by the Birum reference. Specifically, claim 1 requires a number of items that work together in order to provide for the dynamic management of non-volatile memory items. Nothing within the Birum reference teaches the upgrading of non-volatile memory items *per se*. Rather, resources are being updated.

In claim 1, as presented, among other things claim 1 requires checking a unique identifier item stored in non-volatile memory to see whether it exists, and if it exists, to check a value against a software identifier. The method further requires writing the software identifier to the unique identifier item.

The applicant submits that the Examiner is choosing various items within the Birum reference and then putting them together in a way that the Birum reference does not teach. Specifically, the Examiner has identified the unique identifier item as being related to a resource list in paragraph 22. The paragraph specifically states: "where resources that belong to a particular version of an application are identified and placed in a list

(hereinafter this version is called "V1")". The Examiner is therefore equating the non-volatile memory items with the resource list.

For the checking of the unique identifier item against the software of an application, the Examiner points to paragraph 6 and the sentence quoted by the Examiner when fully enumerated states "a list of resources in a current version of an application is created and compared to a list of resources in a new version". The comparison in this case is therefore a list of resources for one version against a list of resources of a second version. Paragraph 9 talks about resource lists again and the resources stored locally.

For the writing of a software identifier to a unique identifier item, the Examiner quotes paragraph 45, which talks about resources in a version that may be untouched by version control. The applicant is unsure how the Examiner came to the conclusion that "writing said software identifier to said unique identifier item" is anticipated by the utilization of a configuration file to leave certain resources untouched by version control. In fact, the applicant submits that the method, as claimed in claim 1, requires the unique identifier item to be changed by the process.

Further, the applicant submits that the various elements chosen throughout the Birum specification could not be put together in a way that would lead a skilled artisan to the invention of claim 1. Specifically, the Examiner has equated the unique identifier with a resource list called "V1". Under the Examiner's construction, claim 1 would require that the process check to see whether a unique identifier exists within the resource list, comparing that item within the resource list with software - paragraph 6 compares two version lists together, not a unique identifier against the application software identifier [i.e. a software identifier located in software on said wireless device as required by the claim]. Further, the method then requires the writing of a software identifier to the unique identifier item. This would, in accordance with the construction above, require that the resource list be updated by writing a software identifier to the unique identifier item after changes are made on the mobile device. Birum simply does not teach this. The configuration file

designation for a resource is simply different from the resource list the Examiner has identified for the same item and equated to the term "**said unique identifier item** in claim 1.

Based on this, the applicant submits that a fair constriction of the Birum reference with reference to claim 1 does not show the method of claim 1.

Similarly, claim 10a requires checking non-volatile memory items to see if a unique identifier item exists, checking the value in the unique identifier item against a software identifier and if the unique identifier item does not exist or if this unique identifier item is different from the software identifier, receiving a set of changes and then writing the software identifier to the unique identifier item. For the reasons above, the applicant submits that the Birum reference simply does not teach this.

Claim 10b similarly requires a microprocessor having means for checking said non-volatile memory items for a unique identifier item and means for updating non-volatile memory; and, claim 10b has been amended to include the writing of the software identifier to the unique identifier item.

The applicant submits that when talking about "said unique identifier item" the Examiner must be consistent, and by doing so, it is not possible to come up with the claim as presented by reference to Birum.

With respect to the Examiner's objection to claim 3, the applicant submits that the writing step being performed after the updating is complete further differentiates the present application from Birum. Specifically, Birum teaches the updating of resources based on resource lists and purge lists but does not teach anything occurring after an update is finished.

With regard to the Examiner's objection to claim 8, the applicant respectfully submits that the updating of an old configuration file into a format compatible with a new application is different from the mapping of non-volatile memory items from one version to the next. While the applicant agrees that in order to modify an old configuration file to be acceptable for a new format, a set of changes are needed by the device in order to ensure that the new configuration file is compatible, the applicant submits that this is different from the mapping as required by claim 8. Mapping, if given a plain meaning, and further reinforced by the disclosure on page 13 in the paragraph under table 4, means directing the application from "an old resource" to "a new resource" in the words of the Birum reference. This is different than changing a resource to be compatible with the new version. The applicant submits that this is therefore also distinct from the Birum reference.

Similarly, claim 5 has now been modified to remove the word "preferably" and therefore the updating step requires the creation of a new non-volatile memory item rather than replacing an existing non-volatile memory item to facilitate roll-back. The applicant submits that this is not presented by Birum. Specifically, Birum requires that overlapping resources are sent to a purge list [for example, see paragraph 25 of the Birum reference].

Claim 6 also requires that the updating does not delete non-volatile memory items that have been previously created. Again, Birum always uses a purge list and therefore is distinct from this claim.

The remaining claims all depend from the independent claims and for the reasons submitted above, the applicant submits that these claims are also distinct from the Birum reference.

Based on the foregoing submissions and amendments, the applicant submits that the application is now in a form acceptable for allowance and reconsideration leading to allowance is respectfully urged.

Should the Examiner wish to discuss any of the matters above with the agent for the applicant, the Examiner is urged to call Joseph Ulvr at 613-232-7302.

Respectfully submitted,

Joseph L. Ulvr
Reg. No. 57696
MOFFAT & CO.
427 Laurier Ave. W., Suite 1200
Ottawa, ON K1R 7Y2
(613) 232-7302

Attorney for Applicant
JLU:jh
Doc. 163246